Simulating Multiple Micro-Aerial Vehicles and a Small Unmanned Aerial Vehicle in Urban Terrain Using MultiUAV2

S. J. Rasmussen,* M. W. Orr,† D. Carlos,‡ A. F. Deglopper,§ B. R. Griffith,¶
Air Vehicles Directorate, Air Force Research Laboratory, Wright-Patterson AFB

Much of modern warfare is conducted in built-up areas. This has created challenges in detecting and tracking potential targets. Because of their size micro-aerial vehicles (MAVs) can be used to get in close to identify and track targets. The ability of MAVs to perform visual surveillance of "over-the-horizon" targets is limited by their range. The MAV's usefulness can be enhanced by releasing them from small uninhabited aerial vehicles (SUAVs) over the urban area. This paper details the components necessary to simulate MAVs being dropped into urban areas, in teams, to perform surveillance on pre-identified targets. The mission is explained. After that we describe MultiUAV2, the MAV and SUAV dynamics models, the mission sensor, and the urban terrain model, and the built-up area wind model.

I. Introduction

The advent of micro-air vehicles (MAVs) and small unmanned aerial vehicles (SUAVs) has made it possible to perform new surveillance missions. Modern warfare has created the need to detect and track potential ground targets in hostile built-up environments. Therefore it is logical to investigate the use of MAVs to detect and track these targets. Because of their size, MAVs have the potential to fly in close enough to objects for operators to identify them. The problem with using MAVs in this manner is they are severely limited by their usable range, i.e. flight times of less than an hour. Since SUAVs can have flight times of many hours, a potential solution is to carry MAVs on board a SUAV and then drop the MAVs in the vicinity of targets, thus providing maximum surveillance time for the MAVs. Investigation and flight test of this concept is the aim of the "Cooperation Operations in UrbaN TERrain" (COUNTER) program being conducted by the Air Force Research Laboratory.

In order to test concepts for COUNTER a research simulation has been constructed that makes it possible to evaluate the use of SUAVs and MAVs in built-up areas. The simulation was constructed by adding new capabilities to an existing simulation, MultiUAV2. MultiUAV2 is multiple vehicle simulation, based on Simulink, MATLAB, and C++, that makes is possible to simulate multiple unmanned air vehicles (UAVs) operating cooperatively to perform set missions. In order to simulate the COUNTER concept vehicle dynamic models, optical mission sensors, urban wind models, and building and street models were added to MultiUAV2. This paper details the components necessary to simulate the COUNTER mission to perform surveillance on pre-identified targets. First the mission is explained. After that we describe MultiUAV2, the MAV dynamics model, the mission sensor, city model, and the urban wind model.

II. COUNTER

A layout of the general concept of operations of the COUNTER concept is shown in Figure 1. The SUAV will be launched from a forward area and fly over a city, town, or other area of interest. The SUAV

*Steven Rasmussen is a General Dynamics Senior Aerospace Engineer. *steven.rasmussen@wpafb.af.mil*
†Matthew Orr is an Research Aerospace Engineer. *Matthew.Orr@wpafb.af.mil*
‡David Carlos is an Developmental Control Systems Engineer. *David.Carlos@wpafb.af.mil*
§Anne Deglopper is Student Trainee (Aerospace Engineer) from the University of Maryland. *Anne.Deglopper@wpafb.af.mil*
¶Barton Griffith is Student Trainee (Aerospace Engineer) from the University of Cincinnati. *Barton.Griffith@wpafb.af.mil*

Report Documentation Page

Form Approved
OMB No. 0704-0188

1. REPORT DATE	2. REPORT TYPE	3. DATES COVERED
2006		**00-00-2006 to 00-00-2006**

4. TITLE AND SUBTITLE	5a. CONTRACT NUMBER
Simulating Multiple Micro-Aerial Vehicles and a Small Unmanned Aerial Vehicle in Urban Terrain Using MultiUAV2	5b. GRANT NUMBER
	5c. PROGRAM ELEMENT NUMBER
6. AUTHOR(S)	5d. PROJECT NUMBER
	5e. TASK NUMBER
	5f. WORK UNIT NUMBER
7. PERFORMING ORGANIZATION NAME(S) AND ADDRESS(ES) **Air Force Research Laboratory,Air Vehicles Directorate,Wright Patterson AFB,OH,45433**	8. PERFORMING ORGANIZATION REPORT NUMBER
9. SPONSORING/MONITORING AGENCY NAME(S) AND ADDRESS(ES)	10. SPONSOR/MONITOR'S ACRONYM(S)
	11. SPONSOR/MONITOR'S REPORT NUMBER(S)

12. DISTRIBUTION/AVAILABILITY STATEMENT
Approved for public release; distribution unlimited

13. SUPPLEMENTARY NOTES
The original document contains color images.

14. ABSTRACT

15. SUBJECT TERMS

16. SECURITY CLASSIFICATION OF:			17. LIMITATION OF ABSTRACT	18. NUMBER OF PAGES	19a. NAME OF RESPONSIBLE PERSON
a. REPORT	b. ABSTRACT	c. THIS PAGE			
unclassified	**unclassified**	**unclassified**		**13**	

will act as the "mother ship" and carry the MAVs.[1] The SUAV will observe the area of interest from an altitude sufficient for survivability and potential targets will be designated through the video imagery transmitted to the human operator on the ground as illustrated in Figure 1. When a positive identification cannot be made of a potential target due to target obscuration or insufficient sensor resolution a MAV is launched to take a closer look at the target. The MAV provides the needed viewing angle and distance to target to provide the operator with positive target identification. The launched MAV is then available for re-tasking to examine other targets of interest within the area of operations. When multiple high-priority targets are sensed in a short period of time multiple MAVs are required to accomplish the verification mission. The MAVs will be recovered if convenient but are considered expendable. The purpose of this staged system is to get time-sensitive information unobtainable by other means.

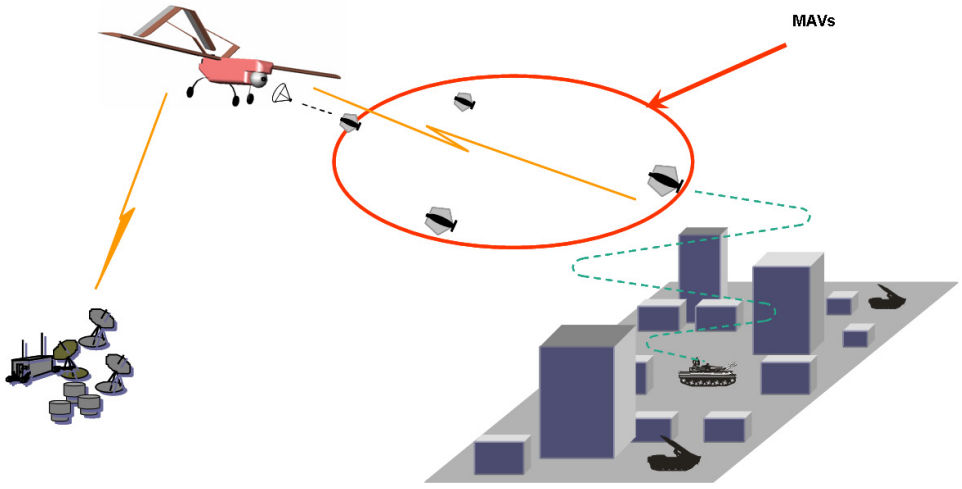

Figure 1. COUNTER concept of operations.

Figure 2 shows the physical rational for this staged MAV/SUAV system. The figure shows a SUAV operating at an altitude needed for sensor coverage of a wide area while placing the SUAV vehicle a survivable distance from the possible target. The distance d_1 from the target to the small UAV is much greater than the distance d_2 from the target to the MAV. The MAV needs to get close to the target in order to get the both the angle needed to view it and get enough pixels on the target to achieve a positive identification. Figure 2 illustrates the point that no matter how good the optical sensor on a high-flying UAV or SUAV is there are still categories of targets that cannot be seen. The obscuration can be caused by manmade means such as buildings, tents, covers, pollution degrading sensor resolution, or smoke. Natural causes could include clouds and dust that render images taken from standoff distances less useful than those from short distances. This heterogeneous system of vehicles will provide a staged team capable of providing persistence, range and the ability to provide positive identification of an obscured target in an urban environment. The SUAV will serve as the communications relay for the MAV and may also provide processing of the MAV video signals. Some of the autonomous decision making/planning capability will be carried aboard the SUAV in the final version of this concept.

III. MultiUAV2

In order to implement and evaluate cooperative control strategies for UAVs, MultiUAV2, a Simulink-based multi-vehicle/multi-agent simulation, was developed.[2] MultiUAV2 is capable of simulating multiple vehicles, targets, and threats. Simulated vehicles include embedded flight software (EFS) and vehicle dynamics. EFS is the software that implements cooperative control algorithms. The vehicle dynamics are simulated with six-degree-of-freedom equations of motion and non-linear aerodynamics. The vehicle model includes an autopilot that makes the vehicles capable of waypoint navigation. An inter-vehicle communication simulation makes it possible to send and record messages between the vehicles.

Integration of the COUNTER mission into MultiUAV2 required changes to MultiUAV2 allowing multiple missions types. MultiUAV2 was developed for the study of cooperative control in a wide area search munition

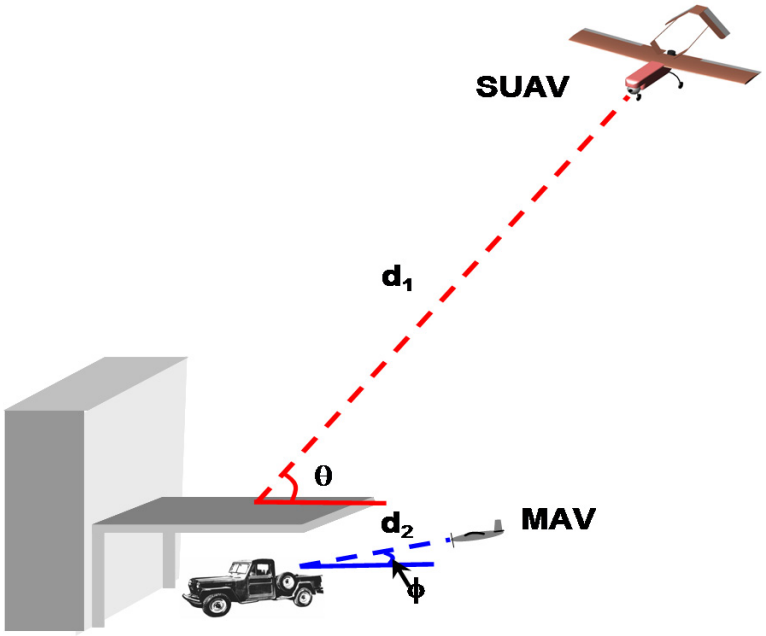

Figure 2. COUNTER sensor placement.

type missions. In order to add the capability to simulate new missions, MATLAB structures were developed that define the missions, including the number and types/capabilities of vehicles need for the mission. To implement new missions, the state transition functions that define the order that tasks are performed were modified to be selectable based on the selected mission. Other changes to Multi$UAV2$ included modifications to task trajectories, addition of multiple initial search patterns, addition of new vehicle dynamics, see §IV, addition of an optical sensor simulation, see §V, and addition of a urban terrain model, see §VI.

IV. Micro-Air/Small Vehicle Simulation Models

COUNTER requires both SUAV Figure 3, and MAV Figure 4 vehicle models. To meet this requirement new vehicle dynamics models were developed by generating new aerodynamic tables and physical parameters which were inserted into Multi$UAV2$'s vehicle dynamics simulation. Vehicle dynamics in Multi$UAV2$ are generated using a simulation called *Variable Configuration Vehicle Simulation* (VCVS).

VCVS is a nonlinear six-degree-of-freedom vehicle simulation that includes a control system which reconfigures the simulation for new aerodynamic and physical vehicle descriptions. VCVS is written entirely in C++ and can run more that 20 times faster than real time. Vehicle dynamics are based on two configuration files, one containing aerodynamic data and the other physical and control system parameters. The aerodynamic configuration file contains tables of non-dimensional forces, moments, and damping derivatives. The vehicle model calculates aerodynamic forces and moments by using the vehicle's state and control deflections as independent variables to look-up values from the aerodynamic tables. During the look-up process, linear interpolation is used for states and deflections not found in the tables. The non-dimensional values obtain from the tables are combined with vehicle state data to calculate forces and moments acting on the center of gravity of the vehicle. These forces and moments are combined with external forces and moments, i.e forces and moments from an engine. The forces and moments are used, along with the physical parameters, to calculate the equations of motion. Included in the model are first-order actuator dynamics, including rate and position limits, and first-order engine dynamics.

By default VCVS uses a control system based on dynamic inversion with control allocation as its inner loop. A rate control system was wrapped around the inner loop to move from angular acceleration commands to angular rate commands. Finally, outer loop controls were added to control altitude, heading, sideslip, and velocity. This was the control system used for the SUAV. For the MAV a different control system was implemented that was more in line with the class of control systems that are implemented on MAVs.

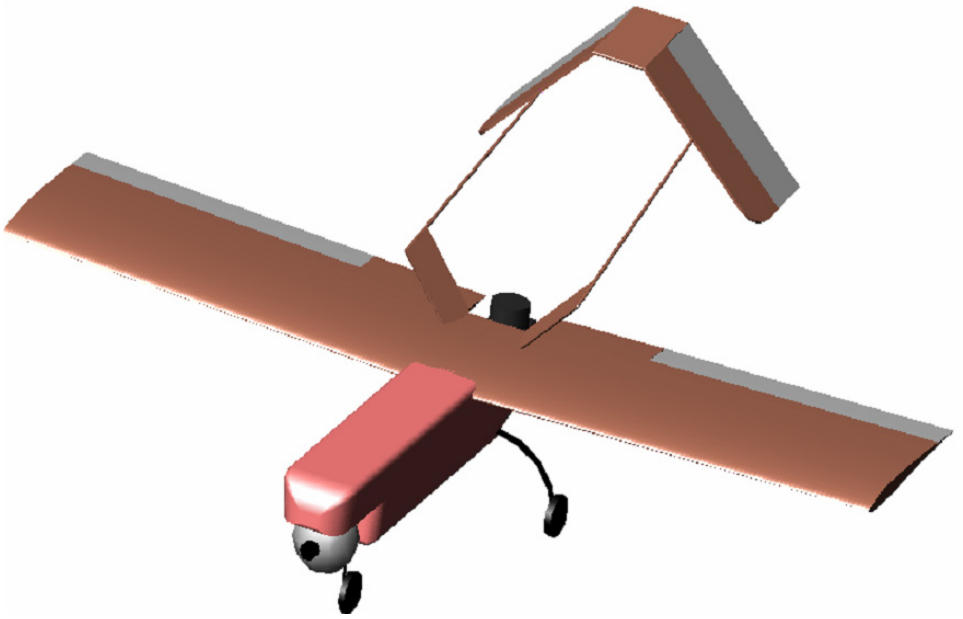

Figure 3. Small UAV.

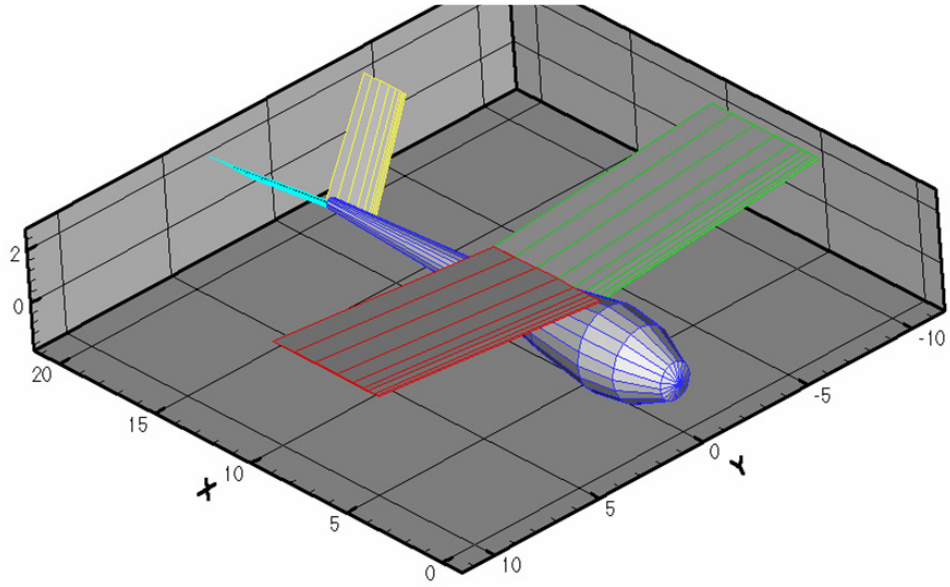

Figure 4. Micro-Air Vehicle

V. Optical Sensor Simulation

As part of the mission, an optical sensor is used to collect and relay images from the MAV to the small UAV and then on to an observer. In this simulation, the process of image collection and identification is simulated through the use of a sensor footprint, attached to the simulated MAV vehicle, a heading dependent template, attached to the target, and stochastic decision rules. For this simulation, the work centered on constructing sensor footprints that represent optical sensors. The heading dependant template, and the stochastic decision rules were based on those already in MultiUAV2. For the COUNTER scenario two types of sensor foot prints were developed, fixed footprints and vehicle state dependant footprints. The fixed footprints have a fixed boundary that moves based on the motion of the vehicle. The vehicle state dependant footprints have boundaries that are calculated at each time step based on the state of the vehicle, i.e. altitude and attitude. The fixed footprints take less time to execute, but are less accurate.

A. Fixed Elliptical Sensor Footprint

A fixed elliptical footprint was developed for use with the MAV simulation. The inputs for the footprint are the aircraft's position and heading angle, the lengths of the major, a, and minor axes, b, of the ellipse, and the offset of the center of the footprint in front of the aircraft, L_{af}, shown in Figure 5. When set to zero, L_{af} indicates that the center of the footprint is directly below the vehicle. Based on these parameters, the positions of the foci of the ellipse, f_1 and f_2, can be calculated. In order to determine if any of the targets are inside the elliptical footprint, the distances between the foci and the target are calculated. By definition, if the sum of these two lengths is less than or equal to the major axis, of the ellipse, then the target is inside the elliptical footprint. For example, given two targets, T_1 and T_2, the distances between the targets and the foci are defined as, L_{1i} and L_{2i}, see Figure 5. For each target, if the equation:

$$L_{1i} + L_{2i} <= a \tag{1}$$

holds, then the target is inside the elliptical sensor footprint.

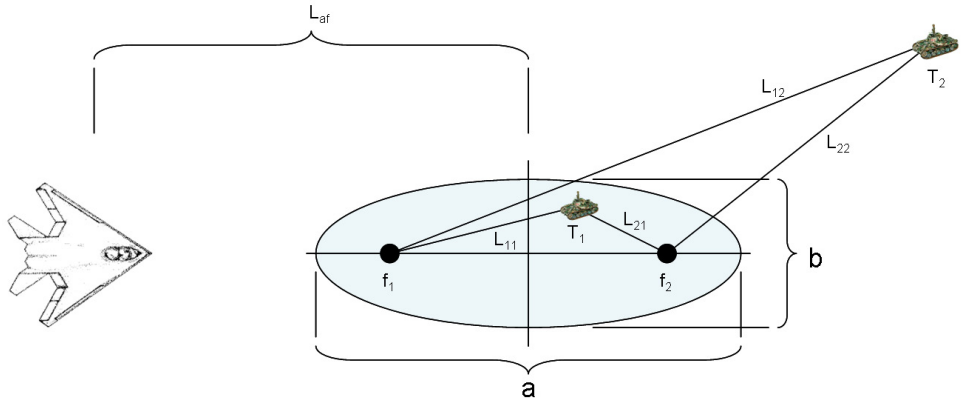

Figure 5. Fixed elliptical footprint.

B. Variable Sensor Footprints

Two vehicle state-dependant variable sensor footprints were developed, one for trapezoidal shapes and one for elliptical shapes.

1. Trapezoidal Sensor Footprint

The trapezoidal sensor footprint is tracked using four rays that extend from each of the four corners of a rectangular sensor face. The intersection of these four vectors with a flat surface creates the four corners of the trapezoidal sensor footprint. Each of the four vectors has an X, Y and Z component in the aircraft's body axes. The Z component has the same magnitude as the camera's altitude. The other two components can be derived from the geometry.

Figures 6 and 7 illustrate the relationship between the X, Y and Z components based on the effective depression angle, α, and the effective horizontal sweep angle of the camera, β. In the figures, one of the four rays, that represent the sensor footprint, is represented as a red line. The point where the upper end of the ray meets the axes is defined as the sensor position. The following equations assume the camera is located at the center of gravity of the vehicle. Using trigonometry the following relationships are derived:

$$Y = \frac{Z}{\tan(\alpha)} \tag{2}$$

and,

$$X = Z \csc(\alpha) \tan(\beta) \tag{3}$$

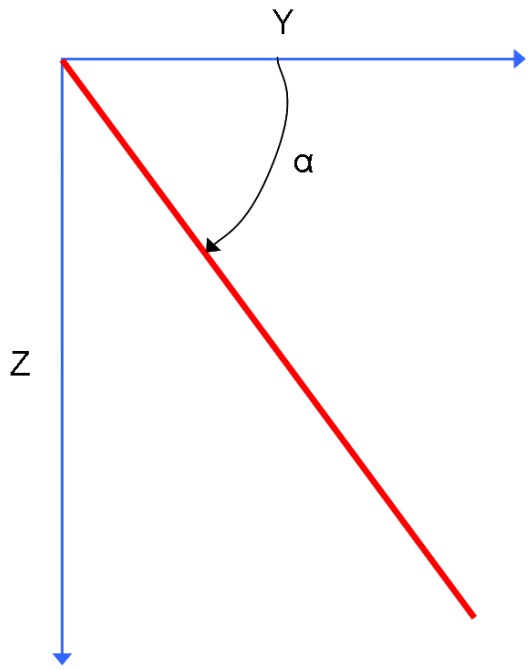

Figure 6. Relationship between Y, Z and α for the trapezoidal sensor footprint

The effective upper depression angle, α_U, or the effective lower depression angle, α_L, see Figure 8, is the sum of the cameras depression angle γ, the camera vertical sweep angle λ, from center of camera to the upper (+) or lower (-) most edge, and the pitch, θ, of the aircraft. The effective horizontal sweep angle, β, is the sum of the cameras horizontal sweep angle plus some additional change due to the bank of the aircraft. The additional change due to the bank makes this angle more difficult to derive. Derivation, yields the following relationship between β and β_{yz}.

$$\beta = \arctan\left[\frac{\tan(\beta_{yz})\sin(\alpha)}{\cos(\theta)}\right] \tag{4}$$

2. Elliptical Sensor Footprint

The elliptical sensor footprint is tracked by the vectors that extend from the camera to each of its four vertices shown in Figure 9. However, if the aircraft has a non-zero bank angle the ellipse will actually be tilted to the left or right and the four points being tracked will no longer intercept the ground at the ellipse vertices. In either case, these four points can be tracked using the equations derived to track the four corners of the trapezoid. The fact that the four points are generally not the vertices of the ellipse makes plotting the ellipse a real challenge. Notice that the center of the ellipse is at the intersection of the two imaginary straight lines connecting points 1 & 3 and 2 & 4. A straight line can be drawn from the camera position through the center of the ellipse and two of its vertices. Using the location of the center (x_c,y_c), the tilt angle of the ellipse, ϕ_e, can be found with the following equation:

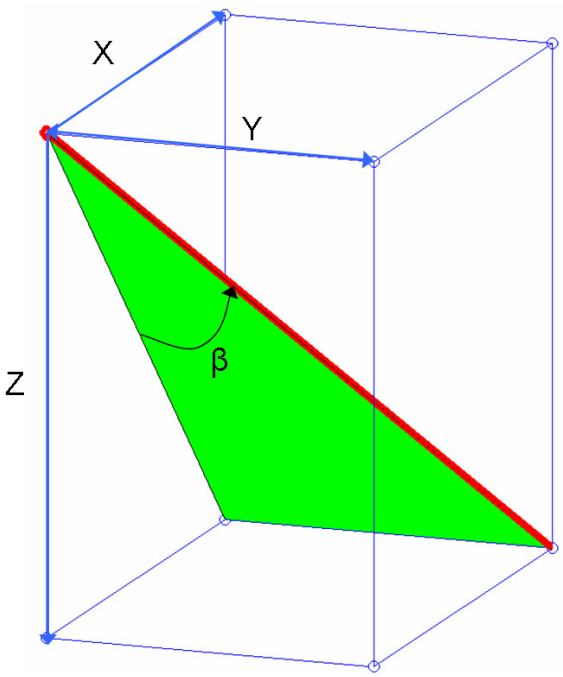

Figure 7. Relationship between X, Y, Z, and β for the trapezoidal sensor footprint

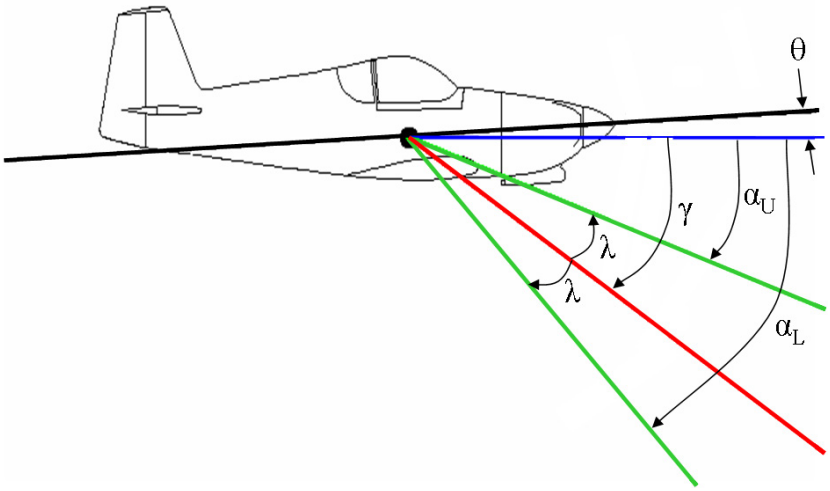

Figure 8. Effective depression angle.

$$\phi_e = \arctan\left(\frac{x_c}{y_c}\right) \tag{5}$$

Once ϕ_e is known, the coordinate system can be rotated so that the X and Y axes parallel the axes of the ellipse.

$$X = x\cos(\phi_e) - y\sin(\phi_e) \tag{6}$$

$$Y = x\sin(\phi_e) + y\cos(\phi_e) \tag{7}$$

Now that the coordinate system parallels the ellipse axes, the following equation can be used to plot the ellipse.

$$X^2 + AY^2 + BX + CY + D = 0 \tag{8}$$

Notice, that there are four constants (A,B,C & D) that are unknown. However, since there are four sets of coordinate pairs that are known, there are four equations which can be solved simultaneously to yield these constants (which define the ellipse). The four coordinate pairs are transformed to the rotated system before the constants are evaluated.

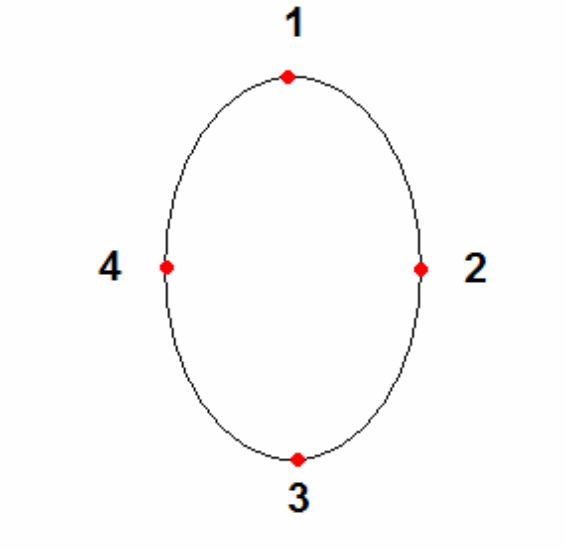

Figure 9. Track points for the elliptical footprint.

VI. Urban Terrain Model

In order to incorporate the effects of urban terrain, MultiUAV2 was modified to accommodate simulation of the effects of buildings and streets. At the very minimum, simulating an urban terrain scenario required simulating blocked lines of sight due to buildings. This was accomplished by requiring that the vehicle and the target are on the same street and the target be in the sensor footprint before the target is reported being in the sensor footprint. To implement the urban terrain, a city model was constructed in a 3-dimensional drawing software package and imported into MultiUAV22 using city management functions. The model includes buildings and streets. At this time, only the streets are used to simulate the city, the building will be incorporated later. To use the city model, utilities were developed to place targets within the city and to manage streets.

The simulated city was designed to include common features of a typical small city. Figure 10 illustrates the location of these particular sections of town. The center of the city contains the layout of a real-world urban operations site. The real-world urban operation site will be used later for live testing. The road layout of the city, with the main roads of the city highlighted in Figure 11, was structured to give common traffic

patterns. In this small city are large parking lots, commonly found near industry or schools, and there are main streets that segment the city.

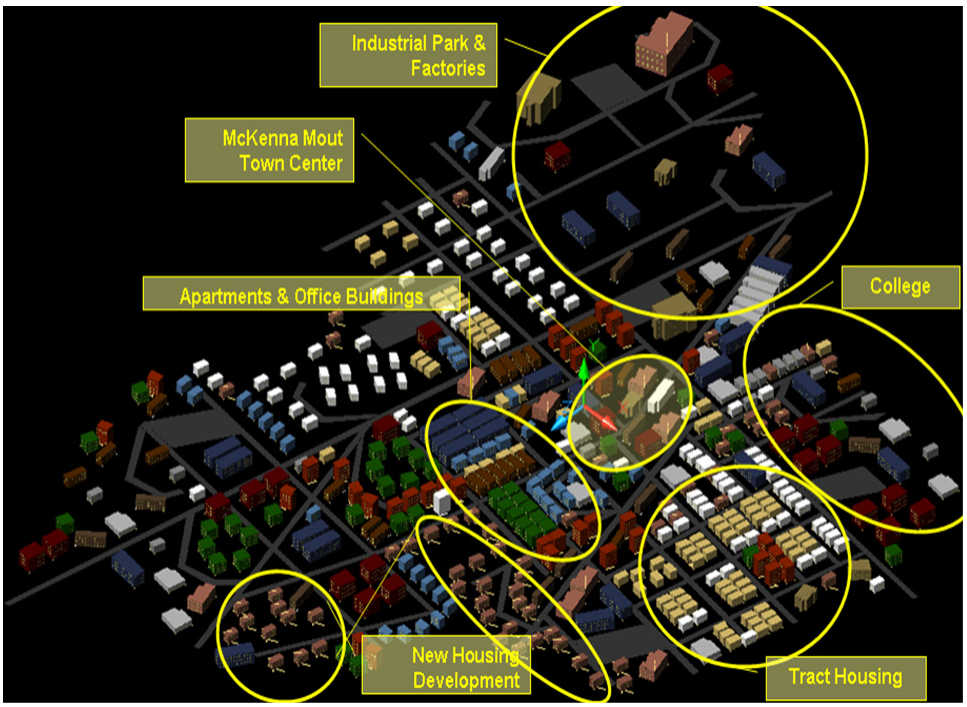

Figure 10. Simulated city.

A. Target Placement Utility

Since the city is large it is difficult to find the precise coordinates to place targets in desired locations. A Graphical User Interface (GUI), shown in Figure 12, accessed from the main menu of the MultiUAV2 simulation, provides target placement functionality. The user initially selects the target position using cross-hairs, after which the target position is shown by a red dot and the coordinates are available in both the legend and in the MATLAB workspace. The target heading is specified by selecting a start point and an end point in the direction of the target. The heading, represented in the utility by a dark red line, is available in the legend and the MATLAB workspace.

B. Road Management Utility

To facilitate the grouping of polygons into roads, there is a GUI which provides a visual click-and-point utility. This utility is accessed from the MultiUAV2 main menu and displays a graphical representation of the City Roads. Using the mouse, individual polygon sections are selected and grouped into single roads for the purpose of the simulation, as shown in Figure 13. When the user is finished, the modified road map is written in the original text file format and in the MAT file format for subsequent uses of MultiUAV2.

C. Simulated City Implementation

At the start of the simulation, the road data is parsed and loaded from text files during the initialization phase. The road map is static and a single instance is available to all the vehicles throughout the duration of the simulation. At the end of the simulation, the road map data is destroyed. If the user chooses to plot the simulation visually, the road data is loaded into MATLAB from a MAT file format and displayed in the figure window. At each time step of the simulation, the sensor footprint functions follow the flowchart illustrated in Figure 14 to determine whether or not the target can be observed from the vehicle. The vehicle queries if the target is located on a road, using a point-in-polygon grid test. If the target is off the road, the vehicle

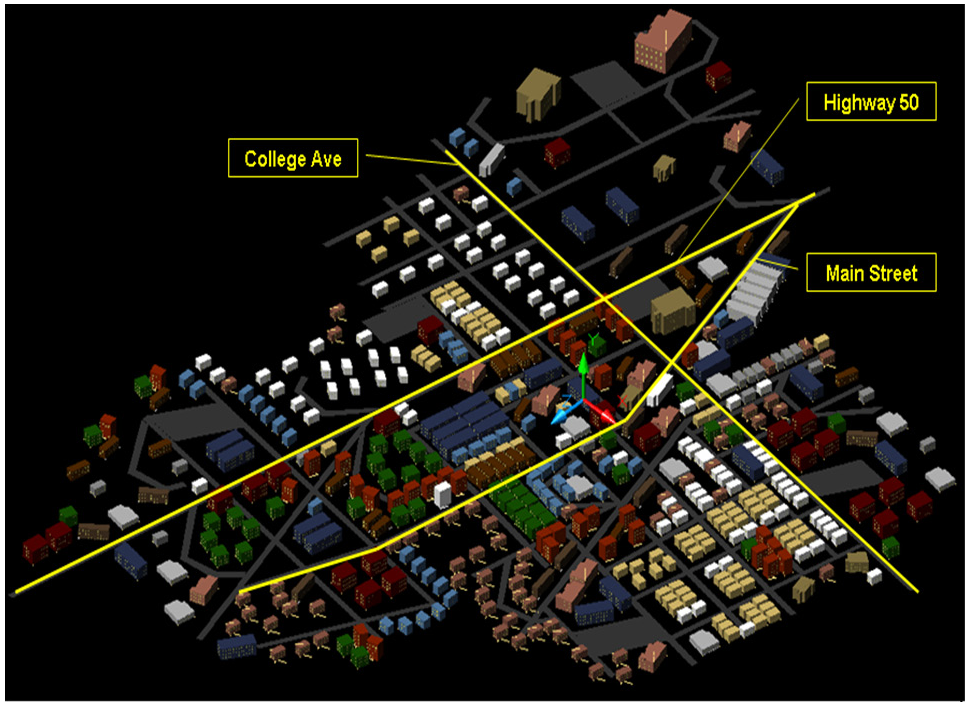

Figure 11. City roads.

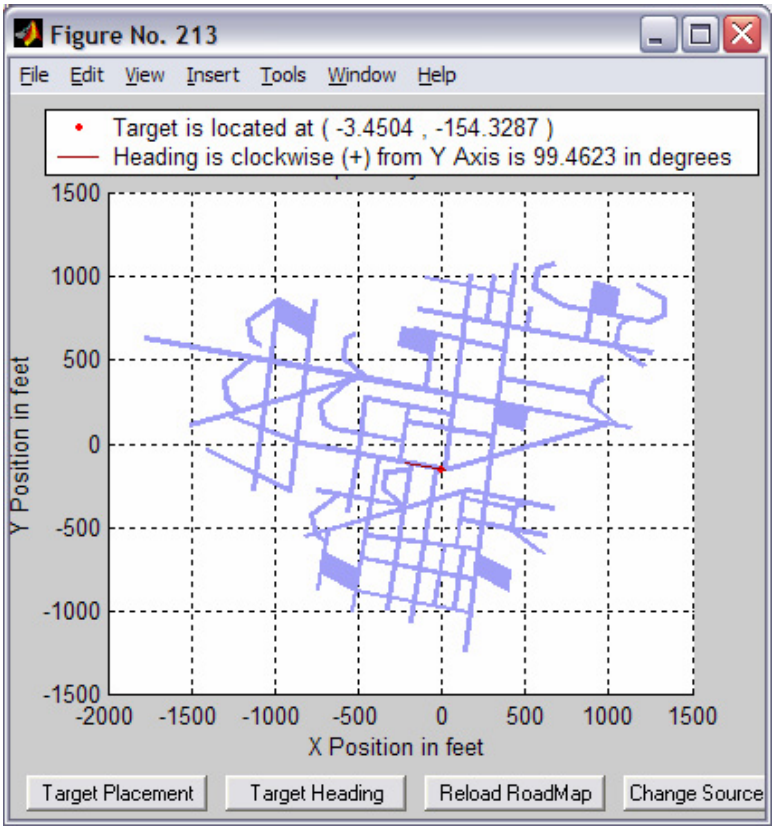

Figure 12. Target placement utility.

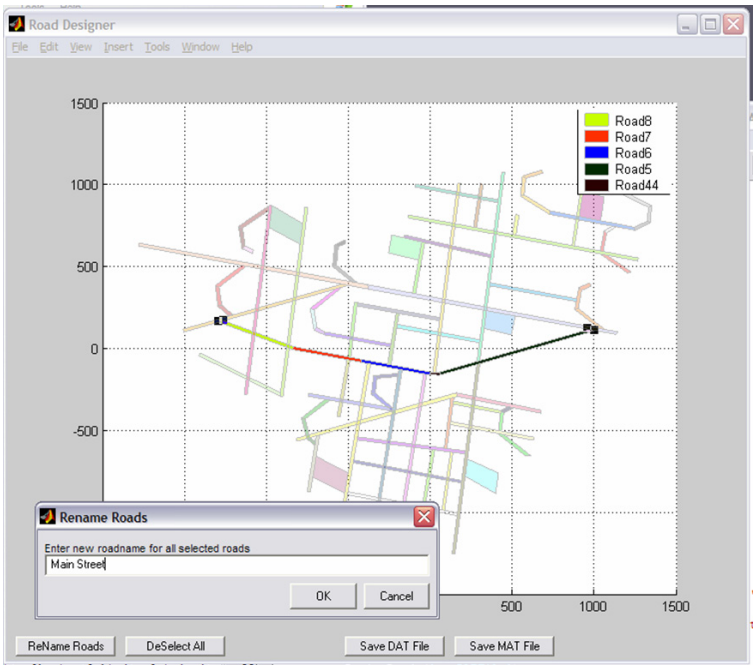

Figure 13. Road management utility.

finds the target if the target is within in the sensor footprint. Otherwise, if the target is determined to be on a road, then the vehicle must determine if the vehicle position is also on the same road.

VII. Wind Model

The wind data for input to the MAV simulation is designed to be read in from a tabular data file. This table contains the X, Y, and Z position and the magnitude and direction of the wind at each point in space. At the start of each simulated time step, an interpolation is performed in order to find the wind value at the vehicle's location in space. These values are passed to the simulation dynamics algorithms and are included in the solution for the new vehicle state.

In order to generate a representative wind field a real-world urban operations site was modelled in a 3D CAD program. This model was then imported into a software package used to build a computation grid around the physical geometry data, which was used to interface with computational fluid dynamics codes. The grid was built by triangulating all the building and exterior domain surfaces. The mesh cells on the building walls were triangles with side length no greater than 12 inches. The entire computational domain was filled with tetrahedrons mapped from the boundary domains. The total grid contained 1,970,076 cells. Using this grid a flow field was resolved and processed by placing normalize grid points at 1 ft spacing throughout the region of interested around the buildings. The u, v, and w components of velocity at each of the normalized grid points were written out to a binary file. This binary file was, in turn, was converted by a MATLAB script for use with MultiUAV2 simulations. The magnitudes of the u and v components of the velocity were then mapped onto a constant altitude plane and shown as vectors. Figures 15 and 16, show the wind vectors and include contour planes to indicate potential lines of constant velocity. The regions occupied by buildings are shown, in the figures, as areas of zero velocity.

VIII. Summary

This paper described the simulation elements that were constructed and integrated into MultiUAV2 to simulate a proposed surveillance mission that uses MAVs and SUAVs in an urban environment. The COUNTER mission takes advantage of the different capabilities of the two vehicles to identity potential targets from higher altitude, with the SUAV, and then send a MAV down to a low altitude to get a closer

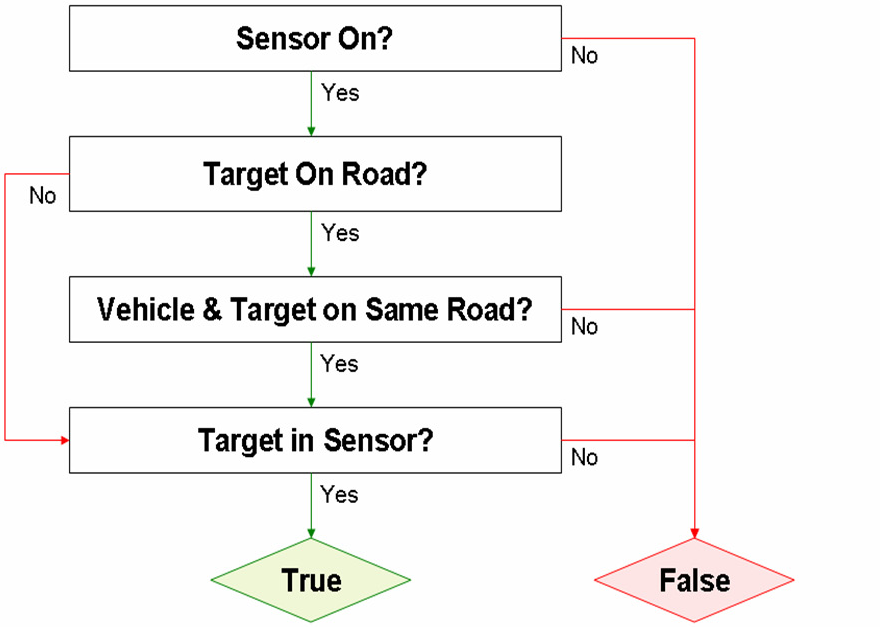

Figure 14. Flowchart for determining whether or not a target can be observed.

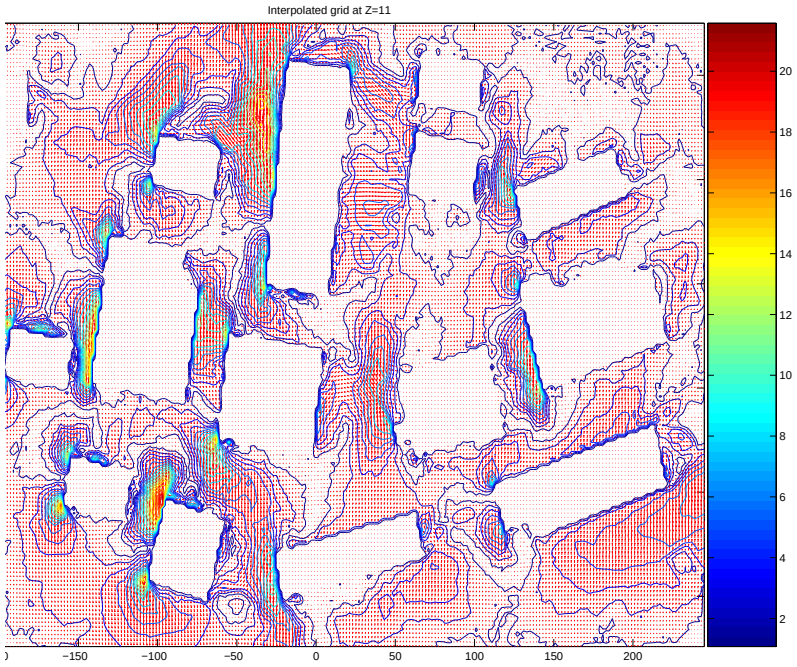

Figure 15. Interpolated values from the wind field data.

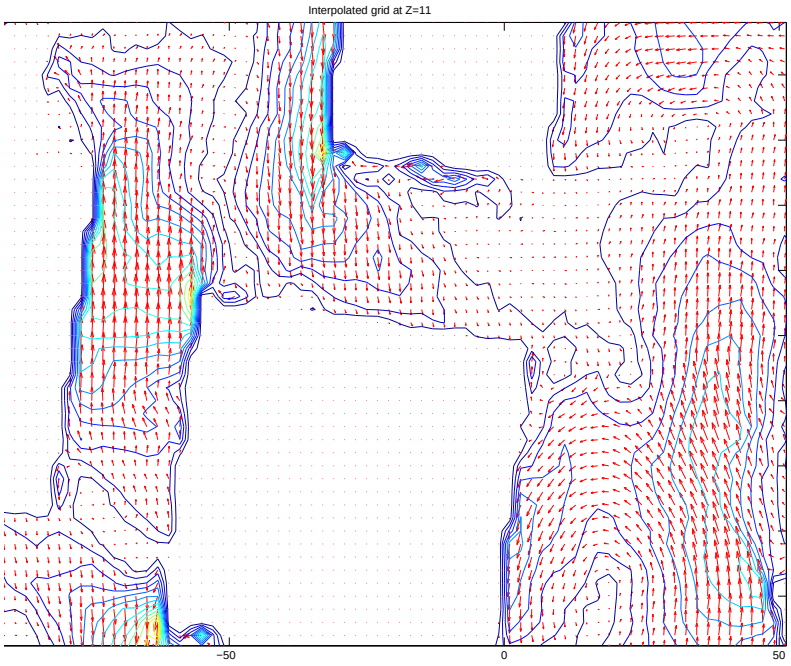

Figure 16. Close up of interpolated wind field data.

view of the target, from a better angle. In order to simulate this mission changes were made and new vehicle dynamics were added to MultiUAV2. Since MultiUAV2 was designed to simulated a wide area search munition scenario, it was necessary to add MATLAB structures to make other missions possible. New vehicle dynamics were developed though the use of vehicle geometries and DATCOM to estimate aerodynamic parameters. The parameters, along with estimated physical parameters are used in MultiUAV2's VCVS dynamics simulation to simulate both the MAVand the SUAV. In order to simulate the optical sensors on both of the vehicles, new sensor footprint algorithms were developed and implemented. To simulated the effects of city buildings on the MAV's ability to view targets, new functionality was added that makes it possible to define city buildings and streets in a 3D CAD packages and then translate the resulting data file into a form that can be used by MultiUAV2. Finally, because of the susceptibility of the MAV to winds, a wind model was incorporated into MultiUAV2 that makes it possible to look-up the three dimensional wind vector for each position of the vehicle.

References

[1]Orr, M. W., Rasmussen, S. J., Karni, E. D., and Blake, W. B., "Framework for Developing and Evaluating MAV Control Algorithms in a Realistic Urban Setting," *Proceedings of the 2005 American Control Conference*, Portland, OR, 2005.

[2]Rasmussen, S. J., Mitchell, J. W., Chandler, P. R., Schumacher, C. J., and Smith, A. L., "Introduction to the MultiUAV2 Simulation and Its Application to Cooperative Control Research," *Proceedings of the 2005 American Control Conference*, Portland, OR, 2005.